#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#define MAX\_FILES 10 // Max number of files in the directory

#define MAX\_FILENAME\_LEN 50 // Max length for file names

// Struct to represent a file entry

typedef struct {

char filename[MAX\_FILENAME\_LEN];

int size; // Size of the file (for example, in bytes)

} File;

// Function prototypes

void listFiles(File directory[], int count);

void addFile(File directory[], int \*count);

void deleteFile(File directory[], int \*count);

void searchFile(File directory[], int count);

int main() {

File directory[MAX\_FILES]; // Array of files (single-level directory)

int fileCount = 0; // Current number of files in the directory

int choice;

while (1) {

printf("\nSingle-Level Directory Menu:\n");

printf("1. List Files\n");

printf("2. Add File\n");

printf("3. Delete File\n");

printf("4. Search File\n");

printf("5. Exit\n");

printf("Enter your choice: ");

scanf("%d", &choice);

switch (choice) {

case 1:

listFiles(directory, fileCount);

break;

case 2:

addFile(directory, &fileCount);

break;

case 3:

deleteFile(directory, &fileCount);

break;

case 4:

searchFile(directory, fileCount);

break;

case 5:

printf("Exiting program...\n");

exit(0);

default:

printf("Invalid choice! Please try again.\n");

}

}

return 0;

}

// Function to list all files in the directory

void listFiles(File directory[], int count) {

if (count == 0) {

printf("No files in the directory.\n");

return;

}

printf("\nFiles in the Directory:\n");

for (int i = 0; i < count; i++) {

printf("File: %s, Size: %d bytes\n", directory[i].filename, directory[i].size);

}

}

// Function to add a new file to the directory

void addFile(File directory[], int \*count) {

if (\*count >= MAX\_FILES) {

printf("Directory is full! Cannot add more files.\n");

return;

}

File newFile;

printf("Enter filename: ");

scanf("%s", newFile.filename);

printf("Enter file size in bytes: ");

scanf("%d", &newFile.size);

directory[\*count] = newFile;

(\*count)++;

printf("File '%s' added successfully.\n", newFile.filename);

}

// Function to delete a file from the directory

void deleteFile(File directory[], int \*count) {

if (\*count == 0) {

printf("No files to delete.\n");

return;

}

char filename[MAX\_FILENAME\_LEN];

printf("Enter filename to delete: ");

scanf("%s", filename);

int index = -1;

for (int i = 0; i < \*count; i++) {

if (strcmp(directory[i].filename, filename) == 0) {

index = i;

break;

}

}

if (index == -1) {

printf("File not found in the directory.\n");

} else {

// Shift the remaining files to fill the gap

for (int i = index; i < \*count - 1; i++) {

directory[i] = directory[i + 1];

}

(\*count)--; // Decrease the file count

printf("File '%s' deleted successfully.\n", filename);

}

}

// Function to search for a file in the directory

void searchFile(File directory[], int count) {

if (count == 0) {

printf("No files in the directory to search.\n");

return;

}

char filename[MAX\_FILENAME\_LEN];

printf("Enter filename to search: ");

scanf("%s", filename);

for (int i = 0; i < count; i++) {

if (strcmp(directory[i].filename, filename) == 0) {

printf("File '%s' found, Size: %d bytes\n", filename, directory[i].size);

return;

}

}

printf("File '%s' not found in the directory.\n", filename);

}

